G I F 4 M O D 4

Version 2

GIF (tm) Image Decoder for the TRS-80 (tm) Model 4 (tm)

4108-C Fairlake Lane, Glen Allen, Va. 23060

TABLE OF CONTENTS

DESCRIPTION

GIF4MOD4 is a decoder for GIF (tm) graphics images for the TRS-80 (tm) Model 4 (tm). It was written entirely in Z-80 assembly language for speed, which helps compensate for the Model 4's (tm) relative slowness.

It decodes Version 87a GIF (tm) graphics files, processes the data, and places the result on the monitor screen if an optional, extra cost high resolution graphics board is installed, or in a "/HR" format disk file if no graphics board is installed, or both on the monitor screen and in a disk file, at the option of the user, if a high resolution board is installed.

If you wish to dump the image to your graphics-capable printer, several software utilities are available for this purpose, both in the public domain and among those which were supplied with your hi-res board.

GIF4MOD4 will decode any image up to 640 x 480 x 256. This means "640 pixels wide" by "480 pixels high" with "256 colors." Hardware limitations preclude decoding larger images. If EITHER the width exceeds 640 pixels OR the height exceeds 480 pixels, the program will abort with an error message. Under versions 87a and 89a, GIF (tm) images may contain a maximum of 256 colors.

CREATING A GIF4MOD4/HR2GIF WORKING DISK:

The very first thing you need to do is to make a backup of your GIF4MOD4/HR2GIF master disk. Then store the original in a safe place, taking the usual precautions to protect magnetic storage media.

If your Model 4 has been expanded to include 360K or larger disk drives, external disk drives and/or a memory upgrade, space will not be a problem for you. You can store and run the GIF4MOD4 programs in any convenient way, so long as BASIC and all GIF4MOD4 programs are available to the system at the same time. Therefore the instructions below do not apply to you.

If you have a "stock" 64K or 128K 2-drive system, space can be a problem. The best solution is to create a dedicated GIF4MOD4 system disk.

If you have a 128K machine, you will want to make use of a MEMDISK or RAMDISK, whenever possible, to hold the GIF (tm) files to be decoded. Decoding from a MEM/RAMDISK is much faster than from a floppy.

Since interlaced files require temporary disk storage space of up to 150K, this puts a limit of about 70K on the size of interlaced GIF (tm) files you can decode with your hardware. This limitation applies ONLY to interlaced images. Also, if you don't have a hi-res board, you'll need another 19.5K on the SAME disk to hold the /HR file to be produced.

To create a dedicated GIF4MOD4 systems disk:

1. Make a backup of a SYSTEM disk, and place it in Drive 0.

2. From DOS Ready, enter: PURGE :0 (I,S)

3. In response to the prompt you will be given for each file on the disk, enter "N" for each /SYS file *EXCEPT* SYS5/SYS, SYS8/SYS, SYS9/SYS and SYS13/SYS, for which you will enter "Y". Enter "N" for each of the three BASIC files, and "Y" for all others. (Note: if you have 128K, also enter "N" for MEMDISK/DCT (or equivalent). If you have a JCL file to set up MEM/RAMDISK, also enter "N" for both that and SYSTEM/JCL).

4. Place a backup copy of your GIF4MOD4 master disk in Drive 1.

5. One after another, enter the commands:

```
BACKUP /CMD:1 :0
BACKUP /OVL:1 :0
BACKUP /BAS:1 :0
BACKUP /JCL:1 :0
```

The disk in Drive 0 is now ready to be used as your GIF4MOD4 working disk. It will have about 70.5K free space, less up to 6K if you have kept MEMDISK/DCT and/or a MEMDISK-creation and SYSTEM/JCL (3.0K, 1.5K and 1.5K respectively).

When decoding large interlaced files, the GIF (tm) file must be on Drive 0, (or on your MEM/RAMDISK, if you have a 128K machine) and you will need a disk with 150K or more free space in Drive 1 to hold the required temporary disk file. Thus the amount of free space on Drive 0 is the size limit for interlaced GIF (tm) files you can decode on your hardware.

OPERATION

GIF4MOD4 will run under TRSDOS 6.x or LS-DOS 6.3. From "DOS Ready," invoke the program with the following syntax:

GIF4MOD4 filename [/ext][:s][:d][+d][+b][+c][+r][+s]

where "filename" is the name of the GIF (tm) image file

"/ext" is the file extension. If no extension is entered, "/GIF" is assumed

":s" is the logical drive upon which the file can be found. If no drive is specified, all drives will be searched sequentially until the file is found

" :d" is the destination drive for the possible temporary disk file necessary to process interlaced GIF (tm) files. This parameter would normally be used only to override the default value

" +d" is a switch which turns on output to a "/HR" format disk file on the default drive, or on the drive specified by the " :d" parameter above. If no high resolution graphics board is installed, output will be only to the disk file. If a high resolution graphics board is installed, output will be to both the monitor screen and to the disk file.

" +b" is a switch to brighten the image. This is primarily useful when you wish to dump the decoded image to your printer

" +c" is a switch to increase the contrast of the image.

" +r" is a switch which invokes an "exponential color ramp." This increases both contrast and brightness of the image.

" +s" ("soften") is a switch to reduce the contrast of an image

The latter four switches are meant to be used one at a time, not in combinations. Each, except "+s", will somewhat reduce detail in the brighter ranges, while "+s" reduces detail in all ranges, although surprisingly little. However, for many images, this slight loss of detail is more than offset by overall improvement in the appearance of the rendered image as a whole.

To decode a file named PICTURE/GIF existing on drive 2, any of the following commands will work:

GIF4MOD4 PICTURE	GIF4MOD4 PICTURE :3
GIF4MOD4 PICTURE/GIF	GIF4MOD4 PICTURE/GIF :3
GIF4MOD4 PICTURE:2	GIF4MOD4 PICTURE:2 :3
GIF4MOD4 PICTURE/GIF:2	GIF4MOD4 PICTURE/GIF:2 :3

Use of the " :d" parameter (:3 in these examples) will force a temporary file to be written to drive 3, this parameter taking precedence over the default drive assignment.

In the case of the first two commands in either column above, if another file named PICTURE/GIF exists on drive 0 or drive 1, it would be the subject file, and not the one on drive 2.

GIF4MOD4 now attempts to locate and open the specified GIF (tm) file. If the file cannot be found, or there is another error, the program will abort with an error message. If the file is successfully opened, certain information is read from it to enable the program to make some initial decisions.

If the image screen size exceeds 640 pixels in width, or the screen height exceeds 480, the program will abort with a "too wide" or "too high" message.

GIF4MOD4 then tests for the presence of a high resolution graphics board. If no board is detected, it will check to see if the " +d" parameter was used. If not, a prompt will be displayed similar to:

```
No high resolution graphics board installed!
   Decode image to a /HR disk file (Y/N)?
```

If you respond by typing "N," the program will terminate. Typing "Y" will allow the program to proceed.

The copyright notice and trademark credit "billboard" will be displayed next. If the GIF (tm) file colors are pure black and white, it has a screen height of 240 pixels, and you have not set any switches which will alter the colors in a pure black and white image, you will be given a message similar to:

```
640 x 240 monochrome image
    No dither required
```

After a short pause, the program will proceed to render the image.

If the file fails any of those tests, a "dithering option menu" will be displayed similar to:

```
Select type of dithering desired:

[0] No dither, use color thresholds
[1] Simple Burkes dither
[2] Burkes cross-dither
[3] Simple Floyd and Steinburg
[4] Floyd & Steinberg cross-dither
[5] Simple Slinkman dither
[6] Slinkman cross-dither
```

Each dithering method will give the image a different appearance. Since each image is different in the number of colors, subject, "texture," contrast, etc., it is likely one method will give a more pleasing rendition than the others.

It is left to the user to determine which of GIF4MOD4's many different combinations of dithering methods and "switch settings" is best, in his judgement, for a particular image. Beauty, after all, is in the eye of the beholder.

See the "Dithering Options" section for more detailed information.

When the user is presented with the above menu, he must press one of the keys

"0" through "6" in order to proceed. Pressing BREAK at this point will terminate the program.

The screen dimensions are now examined again. If the width is neither 320 nor 640, a warning message will be issued similar to:

Non-standard screen size: 572 x 350

Image may be somewhat distorted

and the program will proceed after a short pause.

As far as GIF4MOD4 is concerned, any screen height which cannot be converted to the Model 4's (tm) screen height of 240 through the use of a simple ratio is "non-standard." For example, the CGA image height of 200 can easily be converted to 240 by applying the ratio of 6/5. Similarly, the VGA image height of 480 can be converted by applying the ratio of 1/2. Unfortunately, even though EGA is a standard IBM-compatible screen size, EGA images (640 x 350) do not fall into this category. There is no simple ratio to convert 350 to 240. If the height cannot be converted in exact proportion, a warning message similar to:

Non-standard screen size: 640 x 350

Image may be somewhat distorted

In this particular example, the ratio of 2/3 will be applied, as though the height were 360, giving a less-than-3% size error along the vertical axis, which is unnoticable. Applying the correct ratio (24/35) would not only be extremely time-consuming, storing the 35 lines to convert would require more memory than the program has available to it.

If the screen height is 200, which normally indicates a CGA image, you will be given a prompt similar to:

Image size: 320 x 200
[F]ull Screen or [S]quare pixel?

In GIF (tm) parlance, the term "full screen" refers to an image which has a 4:3 aspect ratio, that is it is 75% as high as it is wide, as is the standard for television screens. This requires a 320 x 200 image to have a 5:6 pixel aspect ratio. However, many CGA images have "square" pixels (1:1 aspect), and will therefore be distorted 20% too high if rendered full screen.

It is suggested you display a never-before-seen image full screen. If it seems distorted, or if you're not sure if it's distorted or not, render it again with square pixels, and compare the results. (See "Dithering Options" and "Demonstration Files" for further information on image distortion.)

Next the various blocks of the GIF (tm) file are examined. If a block not defined in GIF (tm) version 87a is encountered, a warning message similar to:

GIF Function Block type 3 encountered -- not processed.

There will be as many such messages as there are unrecognized blocks. Version 87a has no Function blocks. Version 89a GIF (tm) files have four types of Function blocks, none of which are supported by GIF4MOD4 Version 2.

When the program encounters an image block, it will proceed to decode it.

There are two GIF (tm) image "formats": Normal and "Interlaced." Normally formatted images simply build from left to right, top to bottom.

Interlaced images display in four phases: (1) every eighth line starting at line 0 (the top line); (2) every eighth line starting at line 4; (3) every fourth line starting at line 2; and (4) every second line starting at line 1.

Since the Model 4 (tm) does not have the capacity to store the entire image in RAM (a VGA image needs 640 x 480 = 300K of RAM to store, for example), GIF4MOD4 must store some of the image data to a temporary disk file while it displays the interlaced lines.

This temporary disk file requires 320 bytes times the number of lines in the image. Thus a CGA image (320 x 200 or 640 x 200) needs 320 x 200, or 64,000 bytes (63K), of free space on the destination drive. The largest possible temporary file is 320 x 480 bytes, or 150K.

When an interlaced image is encountered, GIF4MOD4 attempts to create this temporary file. If there is not enough free space on the target drive, the program will abort with an error message.

If adequate free space is found, the process of displaying staggered lines and writing raw data to disk will begin after a brief pause. The fastest writing of temporary files is to a large ramdisk. Next fastest is to a hard drive. Bear this in mind when choosing the default drive.

As soon as the first "phase four" line is encountered, GIF4MOD4 displays the entire image in "normal" order, dithering it properly, reading alternate lines from the disk file and the decoder.

Holding down the BREAK key during decoding will terminate the program and return you to "Dos Ready."

When the rendition of the image is completed, you will see a menu similar to:

[V]iew - toggles image on and off

[Q]uit - exits to Dos Ready

Press "V" and the image will re-appear. Press "V" again, and you will be returned to the above menu. Press "Q" and the program will terminate.

When viewing the image, you may get a more pleasing display by adjusting the Model 4's (tm) contrast control (the rearmost of the two knobs on the computer's under-side to the left of the keyboard).

Because the Model 4 (tm) can only display images in black & white, they can get a bit grainy -- especially the ones with widths of 320 or less. It helps to step back a few feet to view the image.

HR2GIF/CMD

DESCRIPTION:

HR2GIF is a GIF (tm) encoder file conversion utility. It uses Model III (tm) (512 x 192) and Model 4 (tm) (640 x 240) /HR and /HR6 high resolution graphics files, /CHR files (produced by Mel Patrick's "GrafPack" and "FastPack"), and /BLK files produced by Micro-Lab's Pro-DRAW program as input, and creates 640 x 240 x 2 GIF (tm) image files as output.

When encoding 512 x 192 images, HR2GIF does not expand them to the full 640 x 240 screen size, as this could result in undesirable distortion of the image. Instead, it simply centers the image in a 640 x 240 x 2 GIF (tm) screen.

OPERATION:

Invoke HR2GIF from Dos Ready with the following syntax:

HR2GIF filename/ext[:s][:d]

where: "filename" is the name of the hi-res file

"ext" is the file name extension (REQUIRED!)

"s" is the drive where the hi-res file can be found

"d" is the drive to which the GIF (tm) file is to be written.

Legal file extensions are "/HR", "/HR" plus any legal character (e.g., "/HR6"), "/CHR" and "/BLK."

":s" and " :d" are optional.

If you want the GIF (tm) file to be written to any drive other than 0, you must enter the destination drive number (" :d"). Since it is not possible to predict the size of a GIF (tm) file before actual encoding, it is the responsibility of the user to make sure there is enough free space on the destination drive. It is safest to allow 19.5K for the GIF file, although it will virtually always be much smaller than that.

Included in the "billboard" is a prompt similar to:

[N]ormal or [I]nterlaced order?

Type "N" if you want the GIF (tm) file to display in normal order, and "I" if you want it to display in interlaced (staggered) order.

You will then be given a prompt similar to:

[B]lack or [W]hite background?

With 640 x 240 images in "normal" order, you would ordinarily type "B." For images which will not fill the entire screen, such as Model III (tm) 512 x 192 or images from a /BLK file, you may desire a white background. This will have the effect of putting a white instead of black "border" around images smaller than 640 x 240, and/or of having interlaced images "appear" out of a white screen -- an interesting effect.

Next you will be given a prompt similar to:

Version 8[7]a or Version 8[9]a?

Pixels have different shapes on different computers. For this reason, Pixel Aspect Ratio (PAR) information is included in Version 89a files. This tells the GIF (tm) decoder software on the other computer how to preserve the correct proportions of the image. However, many 87a decoders now in place will abort or crash if they encounter a Version 89a file. Thus it is left up to the user to decide whether he wants HR2GIF to produce a Version 89a to tell the decoder about the TRS-80's 1:2 aspect pixels and take the risk that the image will not be displayed at all; or "play it safe" by producing a Version 87a file it may display only half has high as it should be.

Type "7" if you want a Version 87a file with no Pixel Aspect Ratio information. Type "9" if you want a Version 89a file which will contain this information.

If the image is a "/HR" or "/CHR" file, HR2GIF will now proceed to create a GIF (tm) version of it, displaying it as it goes. You will notice the display proceeds at varying rates. The slower the display, the more compression is taking place, except with all-white or all-black areas, when the reverse is true.

If you specified a "/BLK" file, a directory of all image blocks within the file will now be displayed. These are displayed 4 across. If the file contains more than 87 blocks, some will scroll off the screen. This directory is followed by a prompt similar to:

Enter block name EXACTLY as it appears above:

Just as within Pro-DRAW, block names are case-sensitive, and spaces count. Thus if a directory name is "Turtle," entering "turtle" or "TURTLE" will not select the desired block. If the block name you enter is not found, the program will abort with an error message. You can easily re-try, from DOS Ready, by typing <CTL> R (holding down the control key and pressing the "R" key).

When the block is located, you will be prompted to enter the X and Y coordinates of the location on the screen where you want the image to appear. This is consistent with Pro-DRAW's "put" command, i.e., enter the Pro-DRAW X and Y coordinates of the upper-leftmost pixel of the image.

The entered coordinates and the size of the image are now examined. If you entered coordinates which would put all or a portion of the image off the screen, an error message will appear, and the program will abort. Re-try using <CTL> R.

Once legal coordinates have been entered, a GIF (tm) file will be created. This file will have the same name as the "/BLK" file. If you select a block named "Turtle" out of a file named "ANIMALS/BLK," the GIF (tm) file will be named "ANIMALS/GIF." You will probably want to RENAME this file "TURTLE/GIF," especially if you will be encoding several images from the same "/BLK" file.

INCLUDED JCL FILES AND CHECKGIF/BAS

As previously mentioned, there are quite a few "square pixel" files in CGA (320 x 200) format. These images will be distorted 20% too long when displayed in "full screen" mode. Once a file is determined to be a square-pixel file, it can be patched in such a way to force GIF4MOD4 to always display it properly. To accomplish this, the CGA2TRS/JCL file is included. It is invoked as follows:

```
DO CGA2TRS (N="filename")
```

Do NOT enter the "/GIF" extension, as the JCL automatically adds it. Thus, to change a file named AFRICA/GIF, you would enter: DO CGA2TRS (N="AFRICA")

Because GIF4MOD4 gives you the option of viewing CGA files both "square pixel" and "full screen," it was not deemed necessary to provide a JCL file to change files back from "TRS" dimensions to CGA dimensions; so be SURE before you DO CGA2TRS to PERMANENTLY change a file.

Also included is the file FORCEFUL/JCL. When a CGA file is determined to be a full-screen image, you can use this file to patch it to change its height from 200 to 201, forcing GIF4MOD4 to treat it as a full-screen image. This has the side-effect of causing the "non-standard image" message to display, but you may consider that less of an annoyance than having to remember to type the [F] key each time the [F]/[S] prompt appears. Invoke via: DO FORCEFUL (N="filename").

Far less common are square-pixel EGA (640 x 350) files. I have seen exactly two, and both were very old, as GIF (tm) goes (the standard was released in June, 1987). The distortion is very apparent, unlike some CGA files where it's hard to decide whether an image is distorted or not. With square-pixel EGA files, the distortion is nearly 40%; so it's hard to miss.

If you run across an file which causes a "Non standard size: 640 x 350" message to display and looks very stretched out vertically, you can patch the file with EGA2VGA. Invoke via: DO EGA2VGA (N="filename"). If you decide it looks better stretched out, you CAN change it back to it's original form with VGA2EGA/JCL.

Please apply the EGA2VGA and VGA2EGA patches SPARINGLY! Square-pixel EGA files are EXCEEDINGLY rare. I don't want to be blamed if somebody accidentally ruins a perfectly good file and can't figure out how to change it back.

PIKDRIVE/JCL is provided to give you an easy way to patch GIF4MOD4 to change the default for the drive to temporarily store interlaced file data. Typically, those with hard drives or large (150K or more) RAMDISKs will want to change the target drive for faster speed.

Using a backup or "working copy" of GIF4MOD4 (NEVER the original), PIKDRIVE/JCL will change the default setting through the following syntax:

```
DO PIKDRIVE (D=n)
```

where "n" is the number (0 through 7) of the desired new default drive.

CHECKGIF/BAS is a short BASIC program which quickly steps through a GIF (tm) file, and extracts and displays certain information about that file. The information it displays is self-explanatory.

APENDGIF/BAS

APENDGIF/BAS combines files created by HR2GIF into multiple image files. The only limit to now many files can be strung together into a GIF "slide show" is the limit of your disk storage capacity.

When prompted, simply supply the names of the file to be added TO, the file to BE added to the end of the first, and the NEW file to hold the result. Single or multiple image files can be appended to either single or multiple files.

The demonstration file, "BULLY/GIF" is included on your master GIF4MOD4 disk to illustrate some of the things you can do with HR2GIF and APENDGIF/BAS. Run this file through CHECKGIF/BAS and you will see it contains 5 images -- one full-screen interlaced image followed by four much smaller animation frames.

This was originally a line drawing done by my daughter, Pam, when she was nine.

I used Pro-Draw to put the apple in various positions, and save the results to a "/BLK" file, being sure to make a note of the X and Y coordinates of the upper-leftmost pixel of the animation frames. With HR2GIF, I created GIFs (tm) of the 4 animation frames, and then used APENDGIF/BAS to string all five GIFs (tm) into one animated, multi-image GIF (tm) file.

NEGATEHR/CMD

The " +d" switch creates a standard 640 x 240 /HR file of the information which is intended to be displayed on the monitor screen. All white pixels are stored with the corresponding bit set to 1, with black pixels stored as 0. This is the reverse (or "negative," in photographic terms) of information meant to be sent to a printer, where 1 = black and 0 = white.

The program NEGATEHR/CMD will read any Model 4 /HR file, and complement each bit in the file. This permits the altered file to be printed correctly by one of the several public domain file-to-printer routines available.

DITHERING OPTIONS

GIF (tm) images can contain up to 256 colors or greyscales (shades of grey). However, Model 4 (tm) high resolution graphics boards can only display two (black and white). GIF (tm) pixel intensities (brilliance) are stored in one byte, and thus have a range of 0/255ths (off) to 255/255ths (fully on).

Without dithering, display of each pixel on a monochrome display like the Model 4's (tm) would be limited to its color threshold. That is, all intensities in in the range 0 to 127 would be displayed as "black" and all in the range 128 to 255 would be displayed as "white."

Obviously, this method of display will not preserve any of the detail contained in images with more than two colors/greyscales.

The "error dispersion" dithering techniques used by GIF4MOD4 are the latest and most sophisticated methods available to substitute spacial resolution for pixel brilliance.

In addition to the Burkes and Floyd & Steinburg dithering filters, GIF4MOD4 Version 2 introduces the new Slinkman dithering filter. The Burkes and F&S filters were designed for square pixels, but ours are not square.

The Slinkman filter is, to my knowledge, the only dithering method specifically designed for 1:2 aspect ratio pixels (i.e., twice as high as wide). I feel it produces a superior result with images containing 16 or more colors/greyscales.

All these are "two-line" filters, meaning the errors from each pixel are disbursed to its neighboring pixels on both the same line and the next lower line. Each of the three methods can be done either as a "simple" (left-to-right only) or "cross" (left-to-right and right-to-left on alternate lines) dither.

The best dithering method for any particular image depends on the nature of the image. "Texture" is one of the factors which must be considered. The file, BATMAN/GIF, is supplied to demonstrate the effect of texture.

BATMAN/GIF is a 320 x 200 x 4 CGA File with four greyscales.

black	(0/255 = 00H)
dark grey	(85/255 = 55H)
light grey	(170/255 = AAH)
white	(255/255 = FFH)

The background, instead of a solid dark grey, is a pattern of alternating sets of a pair of light grey and a pair of black pixels, thus:

even-numbered lines:	AA AA 00 00 AA AA 00 00 (etc.)
odd-numbered lines:	00 00 AA AA 00 00 AA AA (etc.)

While the average of all background pixels is 55H, the above arrangement gives a much different appearance (texture) than if each pixel had the value of 55H. The best way to see that is to display the image.

To view the image in "raw" form, at DOS Ready, enter GIF4MOD4 BATMAN. Select dither option [0] and [F]ull screen. When decoding is complete, type [V] to put the image back up on the screen.

Notice every 6th line is blank. This is because the image has only 200 lines and was expanded to 240, and there was no dithering module loaded to "manufacture" the missing lines. Notice, too, the image is stretched out vertically. It appears to be a square-pixel image. Press the [Q] key to return to DOS.

Now enter GIF4MOD4 BATMAN again. Select dither option [0] again and [S]quare pixel. Press [V] when decoding is complete. Now the image is in proportion; so it definitely is a square-pixel image. Look at the image. Observe the texture of the background. Note Batman's mask is pure black. Now press [Q] to exit.

Because we now know this is a square-pixel file, we will change it so we won't be bothered with the [F]ull screen/[S]quare pixel prompt again. Enter:

```
DO C6A2TRS (N="BATMAN")
```

Now enter GIF4MOD4 BATMAN again, and select dither option [1]. Notice there is no [F]/[S] prompt this time. When decoding is complete, press V to view it. Looks different, doesn't it? Look at the mask. Dark grey details have appeared which were lost before. Look at the background pattern. It's changed from

```
 on  on off off        to      on  on off off
off off  on  on                off off off  on.
```

Only 3/8ths of the pixels are on. 12.5% of the data has been "clipped" out! Notice the disruption of the patterns below the chin and from the point of his cape to the logo on his shoulder. Also notice his left eye (the one within the mask). There's a small dithering artifact under it. Now press Q to quit.

Enter GIF4MOD4 BATMAN again, dither option [2]. Same ugly background. There's still a pattern disruption under the chin, but different. The artifact under the eye is still there, but smaller, and the pattern disruption from cape-to-logo is gone.

Dither option [3]. Background is correct. No pattern disruption under chin. No artifact under the eye. The only imperfections are some minor pattern disruptions on his suit.

Now decode the image three more times with dither options 4, 5, and 6, and compare the results. I think you'll agree with me that dither option [3], the simple Floyd and Steinburg dither, gave the best result. The Slinkman cross-dither [6] was second best.

Now decode it with the contrast switch set by entering GIF4MOD4 BATMAN +C, and use dither option [5]. I think that's about the best you're going to get, even though the dark grey detail in the mask has been diminished somewhat.

Through this exercise, we have learned two things. First, the "texture" of an image has a great effect on the results produced by the various dithering techniques. Second, you've got to EXPERIMENT! You can't assume anything about an image. Even though this is a simple four-color image for which the simple F&S dither is usually best, and even though the Burkes and Slinkman dithers are usually best on images with a greater number of colors/greyscales, it turned out in this case that the +C [5] combination actually gave the best result.

DEMONSTRATION FILES

The supplied DBEATL/GIF file (640 x 350 x 16 [colors]) is not good quality, but it's one of the two square-pixel EGA files I've ever seen. The other is much better quality, but its subject matter (being "R rated") is not suitable for demonstration purposes.

Decode the image, using dither option [5] (although any except [0] would do).

The image is obviously distorted, being very stretched out along the vertical axis. It's pixels are meant to be square, not standard roughly 3:4 EGA pixels. It's stretched to 240 lines instead of the 175 it should be, a 37% distortion.

From DOS Ready, enter: DO EGA2VGA (N="DBEATL")

Now decode the image again. The quality is still poor, but at least the image is in proportion.

The supplied BWLADY/GIF file is 640 x 200 (CGA) and, while I consider it very good quality, it is rather stark. Decode it first with no switches, [F]ull screen and dither option [5]. Then decode it again with "+s" [F] [5], and observe the changes. The black areas are now a very dark grey, and the stark white area on the subject's face are now a very light grey. Very little detail has been lost, yet some of the abrupt shading changes have been pleasantly "smoothed out." In my opinion, the "+s" switch improves image appearance.

AFRICA/GIF is a 320 x 200 x 256 (color, of which 147 colors are defined) file. It is included for two reasons: first, it is in interlaced format and, second, it provides a clue as to how to decide if a CGA image is square-pixel or not. When decoding, remember it is in interlaced format; so you will need 63K free space on the destination drive. Select dither option [5], and [F]ull Screen. Overall, the image looks reasonably well proportioned. But look at the sun! It's not round, it's an ellipse -- too long in the vertical plane. When you see things in a CGA image that are the wrong shape -- an ellipse which should be a circle, a rectangle which should be a square, etc., it tells you the image is distorted. Decode the image again as a [S]quare pixel image, and you will see what it's supposed to look like. This is also a good image to test the " +s" switch on.

MODTRAIN/GIF is another interlaced CGA file. It demonstrates how difficult it can be with some CGA images to decide whether it's distorted or not. In my opinion, this is a full-screen image -- but that's only MY opinion.

FANTASIA/GIF is a 640 x 400 x 16 (color) file, a fairly unusual format. It is included mainly because I think it's a great image. I recommend decoding with no switches and the simple Slinkman dither ([5]). You may wish to compare the way each dither renders the very dark, but not black, night sky.

Other GIF (tm) files are included, as disk space permits, for your enjoyment.